Journal of King Saud University – Computer and Information Sciences (2017) **29**, 2–18

King Saud University

**Journal of King Saud University – Computer and Information Sciences**

www.ksu.edu.sa
www.sciencedirect.com

# Detection and visualization of non-linear structures in large datasets using Exploratory Projection Pursuit Laboratory (EPP-Lab) software

**Souad Larabi Marie-Sainte**

*Information Technology Department, College of Computer and Information Sciences, King Saud University, Saudi Arabia*

Received 5 June 2015; revised 28 March 2016; accepted 18 April 2016
Available online 11 May 2016

**KEYWORDS**

Exploratory Projection Pursuit;
Genetic Algorithm;
Particle Swarm Optimization;
Tribes;
Clustering;
Outliers

**Abstract** This article consists of using biologically inspired algorithms in order to detect potentially interesting structures in large and multidimensional data sets. Data exploration and the detection of interesting structures are based on the use of Projection Pursuit that involves the definition and the optimization of an index associated with each direction or projection. The optimization of a projection index should provide a set of multiple optima that is expected to correspond to interesting graphical representations in low dimensional space. The implementation of the bio-inspired algorithms along with the projection pursuit develops a new software called EPP-Lab. Projection pursuit is widely used in different scientific domains (biology, pharmacy, bioinformatics, biometry, etc) but not widely present in the well-known softwares. EPP-Lab is dedicated to recognize and visualize clusters and outlying observations on one dimension from high dimensional and multivariate data sets. It includes different statistical techniques for results analysis. It provides several features and gives the user the option to adjust the parameters of the selected bio-inspired methods or to use defaults values. EPP-Lab is a unique software for detection, visualization and analysis of non-linear structures. The performance of this tool has been validated by testing different real and simulated data sets.


## 1. Introduction

In computer science and especially Artificial Intelligence, metaheuristic is a technique intended to find an approximate solution for hard and combinatorial optimization problems in a reasonable time. Metaheuristic methods include nature-inspired by social behavior (Particle Swarm Optimization, Ants Colony Optimization, Bee Colony, Firefly, etc) or by Darwin evolutionary biology (Genetic Algorithms, Genetic programming, Evolution strategies, etc). Most of them solve hard and combinatorial problems (Sevkli et al., 2014; Goswami and Mandal, 2013; Upadhyay et al., 2014) by handling a set of solutions and maintaining a balance between diversification (exploration of the solution space) and intensification (exploitation of the accumulated knowledge). This research work concerns the

Peer review under responsibility of King Saud University.

ELSEVIER Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.jksuci.2016.04.003
1319-1578 

use of biologically inspired algorithms to find out potential interesting structures in large and multidimensional data sets. Extracting hidden information from data of large dimensions involves employing exploratory data analysis methods. A Principal Component Analysis is one known method of statistical analysis that is based on projecting the data on dimensions that maximize the dispersion of the observations. However, maximizing the dispersion does not constantly lead to the detection of interesting structures. In this study, Projection Pursuit (PP), one of Exploratory Projection Pursuit methods, is used. It consists of finding interesting low dimensional projections of high dimensional multivariate data (Jones and Sibson, 1987; Huber, 1985). Based on human visualization, low dimension means (one- two- three) dimensions. PP focuses on the definition and the optimization of an index associated with each direction or projection space. To optimize the projection indices, exact optimization methods (Newton, Steepest ascent, etc) were applied. These methods require the properties of regularity that most of the projection indices do not provide. On the other hand, the main characteristic required for PP is to find multiple optima corresponding to different interesting structures (Friedman, 1987; Morton, 1989; Sun, 1993). However, these exact methods cannot provide multiple optima. Hence, the use of bio-inspired algorithms is encouraging. They can, not only find a global optimum (approximate solution) but also several local optima (using several runs) corresponding to different potential interesting projections. Among the different bio-inspired algorithms, Genetic Algorithm (GA), Particle Swarm Optimization (PSO) and a hybrid Particle Swarm Optimization method called Tribes are employed. The performance of these selected methods combined with PP has been validated in Berro et al. (2010) and Mari-Sainte et al. (2010).

This study is focused on the detection of clusters and outlying observations. Clustering is one of the main tasks of data mining and mainly importuned in different domains (Alghamdi et al., 2014; Aljumah et al., 2013). Outlier detection involves removing anomalous observations from data (Hogge and Austin, 2004). Outliers occur due to mechanical faults, fraudulent behavior, human error, device fault or natural deviations in populations. Their detection can eliminate contaminating effect on the data set.

Although PP has been used for this purpose in different domains (biology, bioinformatics, image processing, biometry, etc), its implementation is not satisfactory (Caussinus and Ruiz-Gazen, 2009).

Therefore, having a software including PP has become indispensable in many scientific disciplines that use this statistical analysis method. In this article, one-dimensional projection pursuit method, including five projection indices, is implemented along with the selected bio-inspired methods in order to obtain a powerful tool called Exploratory Projection Pursuit Laboratory (EPP-Lab).

EPP-Lab is dedicated to look for hidden non-linear structures in high dimension data sets, particularly clusters and outlying observation. This software is designed with collaboration of statisticians. It gives the user the option to adjust the parameters of the bio-inspired methods implemented or to use the defaults values. In addition, it provides new ways for the analysis of the results. EPP-Lab is a unique tool for clusters-outliers detection, visualization and analysis.

To validate the performance of this software, several real and simulated data sets are treated, and some data sets are large and with complex structures.

The rest of this paper is organized as follows. Section 2 introduces the principle notion and definition of PP. Section 3 presents the related works. Section 4 addresses the methodology of the implemented techniques. Section 5 gives a global presentation of EPP-Lab and its main features. Section 6 illustrates the EPP-Lab application and results using several real and simulated data sets to determine clusters and detect outliers, in addition to a small part of comparison studies along with the convergence study. Section 7 tackles the computation time and the limitation of EPP-Lab. Finally Section 8 concludes this research work.

## 2. Projection Pursuit

PP method seeks to look for low- (one-, two-, three-) dimensional projections that provide potential interesting structures hidden in multidimensional and large data sets. The notion of "interesting" structures is defined by a suitable projection index function $I(a)$, depending on a normalized projection vector $a$. This index tries to find the degree of nonlinear structure present in the projected distribution. Let denote by $X : N \times P$ the data set matrix of $N$ cases and $P$ variables. Let $X_i$ is the $i$th column vector in $\Re^p$ associated with the $i$th observation. This study focuses on one-dimensional projection. The projection vector can be defined from $\Re^p$ to $\Re$ as $z = Xa, a$ is a $P$-dimensional vector defining the linear transformation, and $z$ is a $N$-dimensional vector corresponding to the coordinates of the projected observations. So, determining a projection is equivalent to determining $a$. In other words, this matter is equivalent to optimize a selected projection index.

The present work focuses on four particular one-dimensional indices. The Friedman–Tukey index (Friedman and Tukey, 1974) is the first index proposed in the context of EPP and it is interesting for the detection of outliers. The Friedman's index (Friedman, 1987) belongs to the family of the polynomial-based indices, it performs particularly well in detecting separations or clusters compared with other indices from the same family (Sun, 1993). The kurtosis index is based on the fourth moment and has been studied in Peña and Prieto (2001) and Achard et al. (2004). We also consider a new proposal suited to the detection of clusters called "discriminant index" (Berro et al., 2010). All these indices are well defined and applied for the detection of clusters and/or outliers in Berro et al. (2010) and Mari-Sainte et al. (2010).

One of the most important characteristic of PP method is sphering the data. The sphering step is generally applied to data before the PP step. It consists of eliminating scale and correlation structure in the data sets in order to find other aspects of the data. It also ensures the difference between any structures found by PP and those found by Principal Components Analysis. This characteristic is implemented in EPP-Lab.

PP is less widely used compared with PCA but is more powerful than PCA in many cases because PCA only considers the second order moment and may miss interesting hidden structures that can be easily discovered by another EPP technique (Jones and Sibson, 1987). Indeed, important aspects of the data structure are likely to appear in none of the principal subspaces as this may be seen in Fig. 1. Suppose that the whole dimension is larger than 2 and the straight lines are parallel to subspaces of dimension 2. The first principal plane of PCA, roughly the horizontal axis in this example, is clearly unable

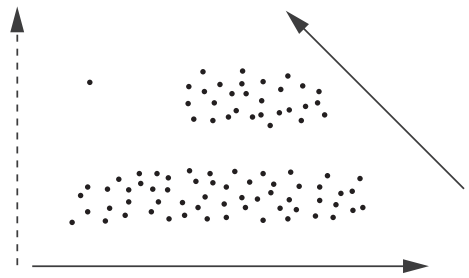

**Figure 1** An illustration of EPP and PCA.

to reveal the two clusters. These clusters could only appear by chance on further projections, and outliers cannot appear on any of the principal two-dimensional subspaces but only by projecting on the oblique line.

## 3. Related works

Optimization procedures for PP have been developed by several authors. Friedman and Tukey (1974) optimized their projection index using the Hill climbing optimization method. However, the projected data changed smoothly with projection direction due to the flatness disadvantage of "Hill clumbing" (Jones and Sibson, 1987). Friedman (1987) used the steepest ascent and the quasi-Newton optimization methods; and (Morton, 1989; Sun, 1993) modified the Friedman algorithm. Peña and Prieto (2001) proposed to use two optimization methods: a modified version of the Newton's algorithm and an optimization method based on the first-order optimality conditions. However, they showed that if the projections were computed for only one direction, then some clusters might mask the presence of others. For other projection indices such as the ones proposed by Morton (1989); Yenyukov (1989); Nason (1992); Posse (1995a); Posse (1995b), their suggested methods are also based on gradient information (steepest ascent, conjugate gradient methods, etc), while (Crawford, 1991; Achard et al., 2004) used heuristic optimization methods.

As highlighted above, the main goals of PP methods are the detection and the visualization of several interesting projections associated with different local optima of a projection index. Most of existing PP optimization algorithms do not find more than one optimum. Therefore, different alternatives have been proposed to achieve this purpose. One alternative focused on the optimization of the projection indices in successive orthogonal subspaces as in PCA (used in Peña and Prieto (2001)). In fact, after finding an optimum for the projection index, the index is re-optimized in the orthogonal subspace of the direction of the initial optimum and so on. More solutions in the initial space (oblique projections) may be interesting but unfortunately they may be missed. Another alternative called structure removal consists of applying a transformation to the data that removes the structure present in the solution (projection) while preserving the multivariate structure that is not captured by it. The PP optimization algorithm is then applied to these transformed data to find additional views that may yield further insight. However, the transformation of the data is performed each time a solution is found, which makes this alternative time consuming. This idea was first used by Friedman (1987), and then employed by Morton (1989) and Sun (1993).

PP is used to solve real problems such as detecting outliers in the field of pharmaceutical trials (Baker, 1991); relating soil patterns to vegetation patterns in ecology (Clements and Jones, 1991); hyperspectral imagery (Achard et al., 2004; Malpika et al., 2008), bioinformatics (Faith and Brockway, 2006) and biometric identification (Ghodami and Larabi, 2015). So, such a method still needs to be developed (Caussinus and Ruiz-Gazen, 2009) when noticing the few number of available softwares.

For the existing softwares, Jones and Sibson (1987) proposed procedures written in Fortran to calculate the univariate and bivariate improved index of Friedman and Tukey (1974). Friedman (1987) also proposed a software written in Fortran. Nason (1992) wrote an Splus function to calculate the three dimensions index he proposed. All these procedures are available on the web site of Guy Nason about PP. Jiayang Sun also implemented the Friedman index in Fortran and C and her program can be installed as a library for S-Plus. The implementation used a steepest ascent approach and the structure removal proposed by Friedman. The software is called Interactive Projection Pursuit and is available on his web site (see Caussinus and Ruiz-Gazen (2009)). PP technique is also present in Matlab since 2001. Martinez and Martinez (2001) developed a computational statistics toolbox in Matlab with some EPP procedures. The index they used is known as the chi-square index and is developed in Posse (1995a,b)). Through a random search, Posse succeeded in determining the global optimum of the projection index and combining it with the structure removal of Friedman (1987) in order to obtain a sequence of interesting 2-dimension projections. Another data analysis tool called Autonomous Projection Mapping (APM) has been developed for the analysis of data from the semiconductor environment (Rohatsch et al., 2006). It implements several indices such as the one proposed by Posse and several optimization algorithms such as genetic algorithms. Several structure removal procedures are implemented: the one proposed by Friedman and the projection on orthogonal subspaces. The interface is written in Delphi but does not seem to be available. All the above-mentioned implementations follow the usual strategy based on iterating a local exact optimization method and structure removal steps. XGobi and GGobi are interactive and dynamic software systems for high-dimensional data visualization where PP is implemented dynamically (Swayne et al., 2003). GGobi is directly derived from XGobi. GGobi contains several projection indices dedicated to reveal different structures (clusters, outliers, holes, ...). The initial starting values are either chosen at random or by the user who looks at the point cloud rotating. So the strategy is different from the one used in the other implementations. The optimization algorithm searches for local optima and there is no need for structure removal. However, for high dimension data sets with complex structure, looking at rotating clouds for a long time in order to discover some potentially interesting starting points may be tedious.

We noticed that the rare existing implementations for EPP are not completely satisfactory. First, most of the optimization algorithms rely on gradient information which imply regularity conditions on the projection indices. Second, the optimization algorithms are usually tuned to get only one "global" optimum (always the same for different starting projections). Moreover, the structure removal may miss some interesting projections or/and is also time consuming. The GGobi/XGobi proposes another strategy but as said previously, the data-analyst has to stay in front of the computer during the whole pursuit

process which also may be very time consuming for high-dimensional data sets. In the following of this study, a different strategy is addressed. We propose a new software including algorithms that do not rely on regularity conditions of the indices and provide different local optima. This research work was part of the author's Ph.D. thesis.

## 4. Projection Pursuit Algorithms based on Bio-inspired Algorithms

PP relies on a projection index and an optimization method called a projection pursuit algorithm. This optimization algorithm should be efficient and flexible in finding out global and local optima susceptible to reveal possible interesting projections at a reasonable time. Evolutionary and biologically inspired algorithms are well known in solving hard and combinatorial optimization problems. In this work, Genetic Algorithms (GA), Particle Swarm Optimization (PSO) and a variant of the Particle Swarm Optimization called Tribes are employed.

### *4.1. Genetic Algorithms*

Genetic Algorithms (GA) are introduced by Holland (1975) and consist of an evolutionary algorithm inspired by Darwinian evolution biology. GA includes inheritance, selection, mutation and crossover.

The population of individuals is randomly generated in order to cover the search space. These individuals must be represented in a specific encoding such as the binary encoding. Each individual is evaluated by means of the fitness function that represents the objective function to be optimized. At each iteration, every individual is evaluated and may be selected and modified (recombined with a possible mutation) to produce a new population. This new population will be used in the next iteration of the algorithm. Usually, the algorithm ends once a maximum iteration number is reached.

In this study we focused on the arguments outlined in Fogel et al. (1966), Rechenberg (1973), Schwefel (1981), and Holland (1975). The initialization of the population is done randomly and the population size is set in the experiment section. Each individual is encoded in real numbers and represents the projection vector defined in Section 2. The fitness function is represented by the projection index also mentioned in Section 2. To select an individual, the tournament selection of 3 participants is applied. Then, the 2-point crossover with $pc = 0.65$ is applied to all the populations. After the selection and the crossover are achieved, a new population of individuals is created either directly copied or produced by crossover. In order to ensure that the individuals are different, a mutation operator is employed to all the individuals with $pm = 0.05$ by choosing randomly one gene and replacing it by a random real value. Algorithm 1 summarizes the proposed GA version. The


**Figure 2** Global presentation of Epp-Lab software.

choices of genetic operator parameters made in this study are based on some experimentations.

**Algorithm 1.** Genetic algorithm pseudocode

**input**: Population size = N;
Max number of iteration = *max_iterations*.
**while** *iteration number* ≤ *max_iterations* **do**
  Evaluate the fitness of each individual.
  Apply a tournament selection of 3 participants.
  Apply genetic operators :
  - Crossover with $p_c = 0.65$
  - Mutation with $p_m = 0.05$

**end**
Determine the best individual

The number of individuals and the number of training iterations are chosen such that the optimum reached cannot be improved easily (by several trials and experimentations). These parameters depend on the data set and on the projection index (Holland, 1975).

### 4.2. Particle Swarm Optimization

Particle Swarm Optimization (PSO) developed by Kennedy and Eberhart (1995) is a population based stochastic optimization technique inspired by social behavior of bird flocking or fish schooling. PSO differs from GA in the way that it has no evolution operators such as crossover and mutation. In PSO, the swarm is composed of potential solutions called particles randomly initialized. Each particle is evaluated by the fitness function to be optimized. The movement of the swarm is directed by the particle itself, the best particle (a particle having the best performance) and the velocity. In other words, at each iteration each particle position is updated according to its best position achieved so far (called *pbest*), the best position of the best particle of the swarm (called global best and denoted by *gbest*) and the velocity. Moreover, the movement of a particle can be affected by its neighbors, especially the best neighbor particle (called the local best and denoted by *lbest*). In this case, the particles move in the search space in close proximity to the local best into the neighborhood set and do not explore the rest of the search space. The definition of the neighborhood depends on the search purpose. Small neighborhoods lead to a slower convergence (to the local best) while large neighborhoods leads to a faster convergence (to the global best). With a global best, the representation of a neighborhood consists of the entire swarm. Furthermore, PSO algorithm needs utilizing some parameters which their manipulation can cause surprising changes in the system's behavior. PSO requires using either the maximum velocity parameter (denoted by Vmax) or the inertia parameter. Vmax is fixed to avoid a rapid moving of particles from one region to the other in the search space. In addition, it prevents explosion and scales the exploration of the particle's search. The inertia factor controls the influence of the velocity obtained in the previous step. A large inertia factor causes a large exploration of the search space while a small inertia factor concentrates the search on a small space.

Many extensions have been suggested to improve PSO algorithm. This study employs the original version introduced by Kennedy and Eberhart (1995) and modified by Clerc (2006). In this implementation, the particle represents the projection vector of $P$ dimensions. The velocity and the particles are randomly initialized. $V_{max}$ parameter is applied $V_{max} = (projection_{max} - projection_{min})/2$ and $V_{min} = -V_{max}$. The fitness function is represented by the projection index as well as in GA. In order to explore simultaneously several regions of the search space to find local optima before finding a global optimum, a new neighborhood version called the Cosine neighborhood is proposed. The idea is to divide the swarm into several groups such that the cosine angle between each projection vector (particle) in the same group should not exceed 30 degrees (for more detail see Berro et al., 2010). In each group, the particle is strictly controlled and moves following its best position (pbest) and that of its group (*lbest*). Once all the particles positions are updated, this structure will be updated by the creation of new groups. Algorithm 2 summarizes the proposed PSO.

**Algorithm 2.** PSO pseudocode

**input**: *getNeighbor*: a function to get a particle's neighbor in the swarm. The number of neighbors = 3;
The number of particles (Swarm.size) is defined according to the data set;
The number of iterations : *max_iterations*;
$C_1 = 0.7$ and *rand*() is a random number uniformly distributed in [0, 1]
*getBestNeighbor*: a function to get a particle's best neighbor (lbest) in the swarm.
*getNeighbor*;
**while** *(iteration number ≤ max_iterations)* **do**
  **while** *(particle number ≤ Swarm.size)* **do**
    Update position: position =position + velocity;
    Calculate the fitness;
    **if** $(fitness > fitness(pbest))$ **then**
      $fitness(pbest) \leftarrow fitness$; $pbest \leftarrow$ new updated position;
    **end**
    $lbest \leftarrow getBestNeighbor$;
    Update velocity: $V \leftarrow C_1 * V + rand()(pbest - position) + rand()(lbest - position)$;
  **end**
**end**

### 4.3. Tribes

Tribes is a hybrid free parameter PSO method developed by Clerc (2006). Tribes method involves a swarm of several sized groups of particles called Tribes. In each Tribe, the particles are interconnected to know the best particle of the tribe (intra-tribe communication). The tribes are linked between them to make a global decision (inter-tribe communication). This structure can be developed and updated through creation, evolution and deletion of particles and tribes. This development depends on measures of quality of tribes (good or bad) and particles (good or neutral) determined through the evaluation of the fitness function. Contrary to PSO, the particles move in the search space through certain moving strategies based on hyperspherical probability distributions, which may be with or without noise, or independent Gaussian. This configuration allows exploring concurrently several promising areas, usually around local optima before making a global decision. For more details, the reader can refer to Mari-Sainte et al. (2010) and Clerc (2006). This method has no parameter to settle, it only requires the definition of a particle (projection vector), the fitness function (the projection index) and also the number of iterations. Algorithm 3 briefly describes the whole method. Note that, position means the current position of a particle, *pbest* is the best position memorized by the particle, and *gbest* is the best position memorized by the best particle of the swarm.

**Algorithm 3.** Tribes pseudocode

**input**: The number of iterations : *max_iterations*;
*L*: *the total number of information links*
1. Create a first tribe formed of a single particle
2. Estimate its fitness (the projection index)
3. Calculate *position* = *pbest* = *gbest*
4. *nb_iteration* + +
5. Create the second tribe, from the first tribe,
6. *L* = 1

**while** *(iteration number* $\leq$ *max_iterations)* **do**
  Estimate fitness of every particle
  Calculate *position*, *pbest*, *gbest* for each particle
  Determine the quality of every particle and every tribe
  **if** *(Number of tribes* < 3*) and (Both tribes did not improve their performance)* **then**
    Create the third tribe, from the first two tribes;
    *L* + +
  **end**
  **else**
    **if** *(iteration number* = $\frac{L}{2}$*)* **then**
      Create a new tribe
      Remove the worst monotribe from the swarm
      Remove the worst particle from each "good" tribe
      *L* + +
    **end**
  **end**
**end**

**Table 1** Italian regions and areas for oils.

| Regions | Areas |
|---|---|
| Southern Italy | Apulia, Calabria and Sicilia |
| Sardinia | Sardinia |
| Northern Italy | Liguria and Umbria |

## 5. Global representation of EPP-Lab

EPP-Lab is a software dedicated to detect hidden non-linear structures in high dimensional data sets. It is designed for handling EPP by using bio-inspired optimization algorithms. It contains three optimization methods (GA, PSO and Tribes) addressed in the above section and five projection indices (Friedman, Friedman Tukey, Discriminant, Maximum and Minimum Kurtosis indices). For more details about the projection indices, the reader can refer to Berro et al. (2010), Ruiz-Gazen et al. (2010), and Mari-Sainte et al. (2010). The code of EPP-Lab is implemented in the JAVA6 language, available on (https://www.researchgate.net/publication or DOI: http://dx.doi.org/10.13140/RG.2.1.4522.2480).

The interface is a combination of graphical and numerical representations. EPP-Lab includes three main windows and each one has its principle role as shown in Fig. 2.

EPP-Lab gives the user the option to projection pursuit or projection analysis. In the first case, the user can choose the "mode" (expert mode offers a wide choice to the user and semi-automatic provides only the choice of parameters), the structure to be looked for (clusters or outlying observations) and can define the data set in addition to the projection index and the optimization method. It gives the user the option to sphere the data and to set parameters. In the second case, EPP-Lab offers the user the option to show projection analysis from a results file. It allows getting directly the analysis window without going through the projection pursuit and the optimization step. Since the results file is saved, it can be analyzed in order to visualize the results at any time, which is a great advantage for this application. The result file (in text format) includes all information about the input (such that the date/time of creating the file, data file name, the data dimension, the selected projection index, the applied optimization method and its parameters, the number of iterations and runs) as well as the output such as the optimum value of the projection index and its associated solution (the provided projection vector) for each iteration and for each run. Note that the input file is a text file containing the data set matrix as described in Section 2 and its dimensions.

The interesting structures obtained after the optimization process is displayed by a graphic. This graphic is addressed using the distribution of the projected data associated with the optimum (or local optimum) for the projection index. It may be a kernel estimator or a histogram. The histogram graphic is built as follows. Once the data are projected, we set an integer value of the minimum and maximum coordinates of the projected data. Then we decide the number of segments to be cut. The length of each segment depends on the number of elements in this segment. For the kernel estimator, we use the triweight kernel (Klinke, 1997) defined in Eq. (1).

**Table 2** Parameters of GA and PSO.

| Methods | Parameters | Small data sets: Lubischew | Large data set: olive, reliability & simulated |
|---|---|---|---|
| GA | Individuals | 50 | 100 |
| | Iterations | 20 | 50 |
| PSO | Particles | 20 | 50 |
| | Iterations | 50 | 100 |
| | Fitness evaluations | 1000 | 5000 |

$$K(z) = \frac{35}{32}(1 - z^2)^3 R_{\{|z| \leqslant 1\}} \quad (1)$$

where $R_\Omega$ denotes the dummy variable associated with the set $\Omega$, and $z$ is the coordinates of the projected data.

In order to analyze several projections and understand the differences between several univariate representations of the data, the projection index values and the cosines of the angles between any two projection vectors are studied. The projections are ordered according to the decreasing values of the projection index and the values of the index are plotted so that the local optima are easily detected. We recall that all the projection indices included in EPP-Lab are to be maximized except the discriminant index and the kurtosis index offered to search for clusters. Without loss of generality, we maximize the minus kurtosis and the minus discriminant indices. The local optima can reveal different structures. However in some cases, different projection index values can correspond to the same structure. The correlation coefficient between two linear combinations associated with both projection vectors can indicate whether these two projection directions yield the same projection or not. This coefficient is equal to the cosine of the angle between these two projection vectors (provided that the data are spherical and the projection vectors are normalized) (Berro et al., 2010). Therefore, the cosines of the angle between the chosen projection vector and the other projection vectors are calculated. If the cosine value is close to 1, these projection directions are likely to provide the same interesting structure. The cosine values are plotted to visualize how far (in terms of angle) two projection directions are.

Furthermore, usually the structure of the data set is quite complex with a large number of classes. Discovering all of them by using one-dimensional EPP is challenging. One possibility offered by EPP-Lab is to identify other interesting projections by splitting the data set in two parts and analyzing each part separately. This strategy is originated by Friedman (1987) called "isolation method" and then applied in various research studies with different designations such that PPTree (Lee et al., 2013).

The second target of this study is outliers detection. EPP-Lab software provides an effective way to achieve this objective. We propose a rule based on k-sigma principle that involves considering an observation as outlying if its distance to the mean (of the projected data) is larger than k times the standard deviation $(k \times \sigma)$ of the projected data. When the value of k increases the number of outliers decreases. This parameter is set by the user and we recommend to be set at least at 3.

## 6. EPP-Lab application and results

The purpose of this section is to show the efficiency of EPP-Lab in the detection of local optima that correspond to projections revealing interesting structures of the data sets. To this, five data sets are tested. The data set is represented by a matrix with $N$ rows (samples) and $P$ columns (variables or features). The five projection indices are employed and optimized using the three optimization methods. In order to well visualize the solutions, a color is assigned to each detected group.

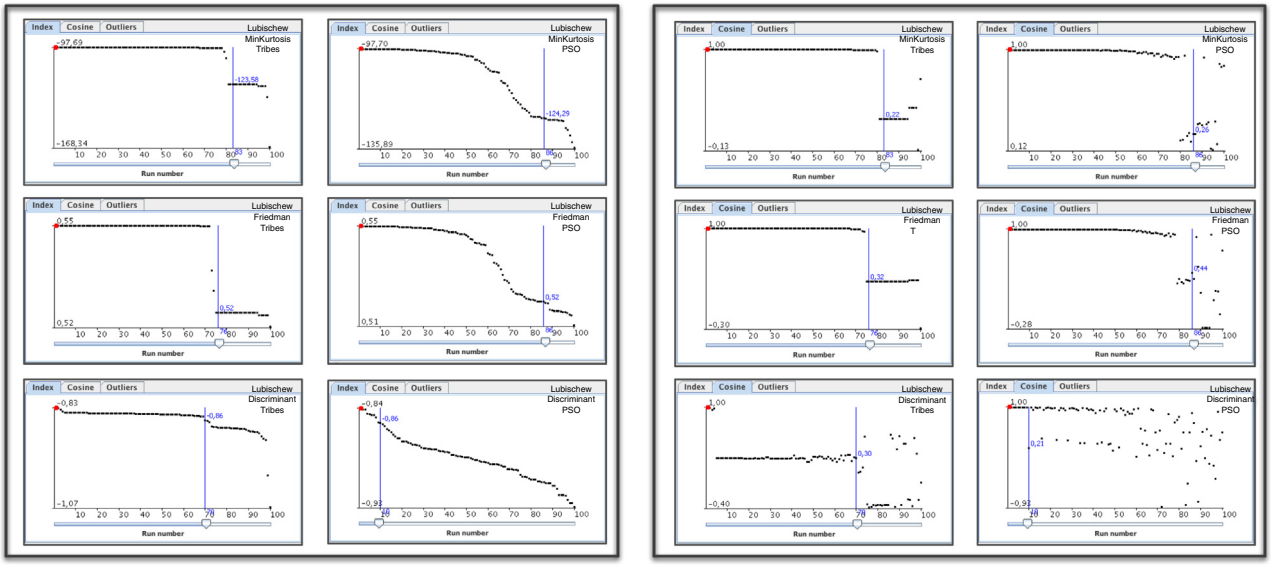


**Figure 3** Lubischew example: plots of the ranked indices (left plot) and the associated cosines of each projection vector with the "best" projection vector (right plot) using Tribes (left of each plot) and PSO (right of each plot) for the minimum kurtosis (top curves), the Friedman (middle curves) and the discriminant indices (bottom curves).

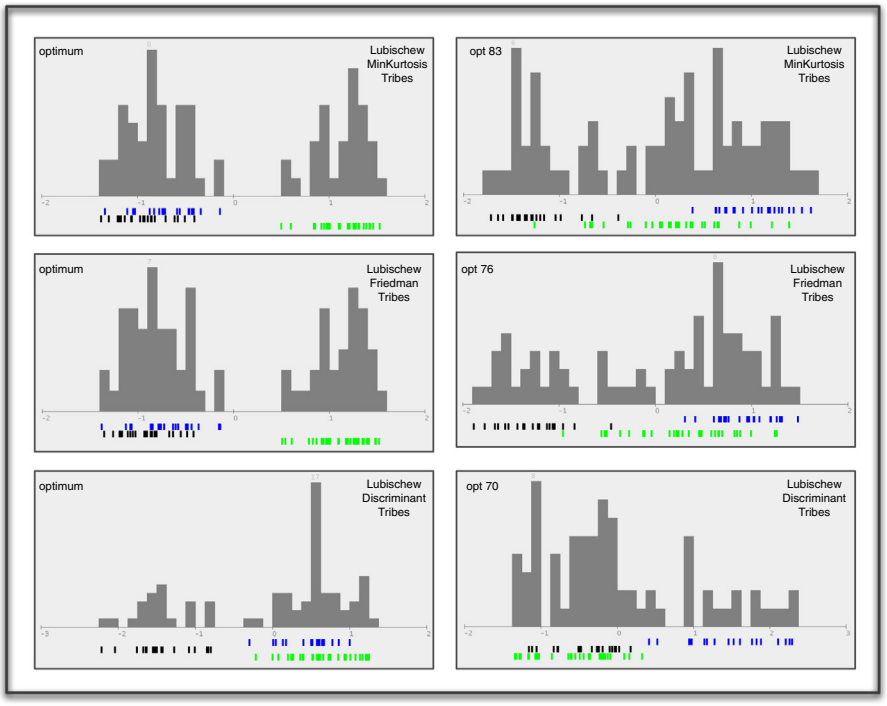


**Figure 4** Lubischew example: histograms of the distribution of the projected data on the "best" projection (left) and on a second "best" projection (right) with Tribes for the minimum kurtosis (top plots), the Friedman (middle plots) and the discriminant indices (bottom plots).

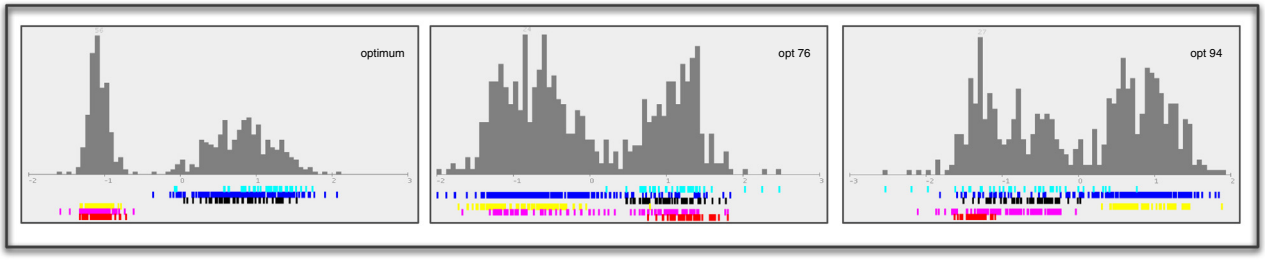


**Figure 5** Olive data: histogram corresponding to the global optimum (left figure) and local optima (middle and right figures) for the kurtosis index using Tribes.

### *6.1. The data sets*

*Lubischew:* it consists of $N = 74$ insects and $P = 6$ morphological measures (Lubischew, 1969) that are structured in 3 clusters. The first one (respectively the second and the third) contains observations 1 to 21 (respectively 22 to 43 and 44 to 74). This data file has already been studied in the context of EPP (see Friedman and Tukey, 1974; Caussinus and Ruiz-Gazen, 2009).

*Olive data:* consists of the percentage composition of $P = 8$ fatty acids found in the lipid fraction of $N = 572$ Italian olive oils. The 572 samples come from 3 different Italian regions (Southern Italy, Sardinia, Northern Italy) subdivided themselves into 6 areas as shown in Table 1. The structure of the data set is quite complex with 6 clusters (see Table 1) which have different shapes in a six-dimensional space.

*Simulated datasets:* we generated two data sets of $N = 1000$ observations and $P = 5$ variables. The observations are distributed according to various mixtures of normal distributions defined as follows:

*Normal4:* contains four clusters of 250 observations with gaussian distribution $\mathcal{N}_5(\mu_i, I_5)$ with $i = 1, \ldots, 4$ where $\mu_1 = (0, \ldots, 0)^T, \mu_2 = (10, 0, \ldots, 0)^T, \mu_3 = (0, 10, 0, 0, 0)^T, \mu_4 = (0, 0, 10, 0, 0)^T$ are 5-dimensional vectors.

*Normal10:* contains ten clusters of 100 observations with gaussian distribution $\mathcal{N}_5(\mu_i, I_5)$ with $i = 1, \ldots, 10$ where $\mu_1 = (0, \ldots, 0)^T, \mu_2 = (10, 0, \ldots, 0)^T, \mu_3 = (0, 10, \ldots, 0)^T, \mu_4 = (0, 0, 10, 0, 0)^T$, $\mu_5 = (0, \ldots, 0, 10)^T, \mu_6 = -\mu_1, \mu_7 = -\mu_2, \mu_8 = -\mu_3, \mu_9 = -\mu_4, \mu_{10} = -\mu_5$ are 5-dimensional vectors.

*Reliability data:* is a real data set from the industry under confidential agreement. It consists of 996 high technology

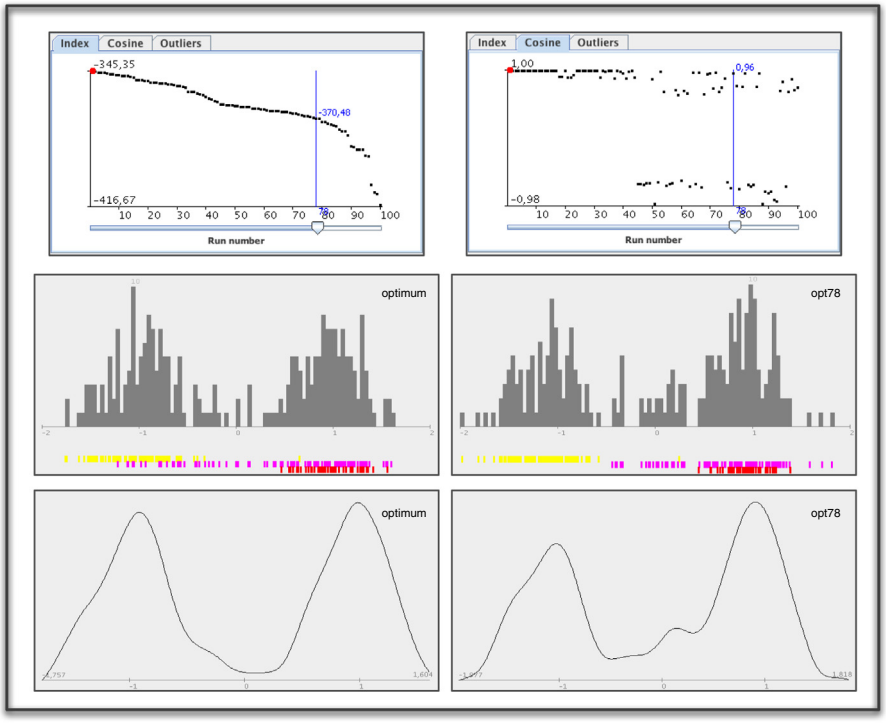


**Figure 6** Olive data: plots of the ranked indices (top left) and the associated cosines (top right). Histogram (resp. kernel density estimator) corresponding to the global optimum (middle left, resp. bottom left) and a local optimum (78th run, middle right, resp. bottom right) for the kurtosis index using GA.

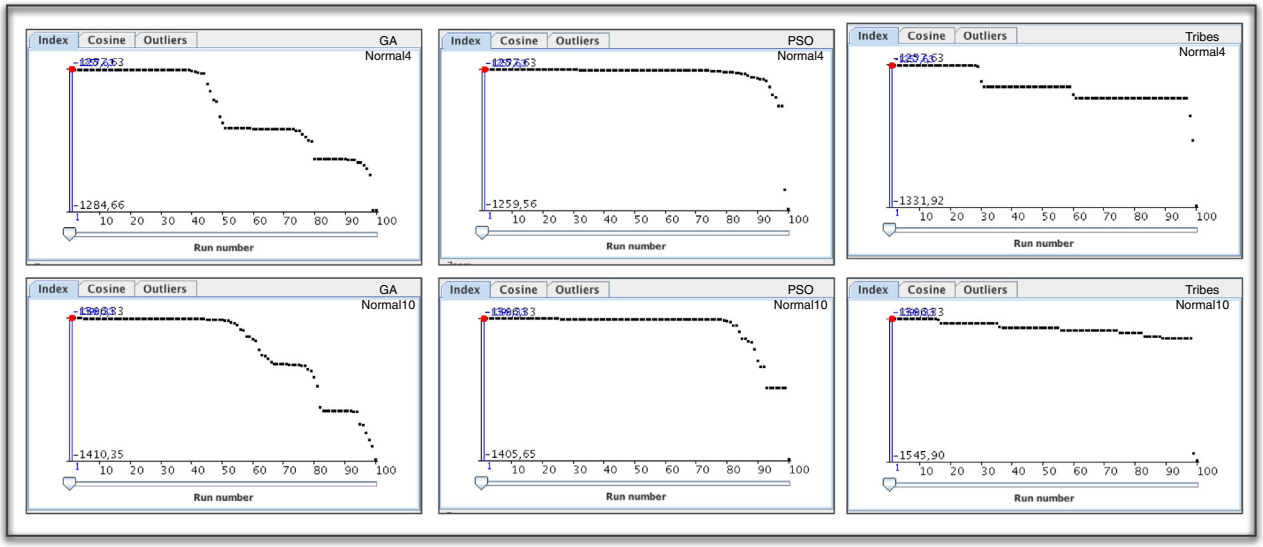


**Figure 7** Simulated data: plots of the ranked values of the minimum kurtosis index for the Normal4 (top curves) and the Normal10 (bottom curves) with GA (left curves), PSO (middle curves) and Tribes (right curves).

chips and 10 variables. The purpose of the analysis is to detect multivariate outlying observations that may represent flawed chips. The chips were sold but there were some problems on the chip number 262.

### *6.2. Results*

GA and PSO require setting some parameters. For small data sets like the Lubischew, the number of individuals for GA and

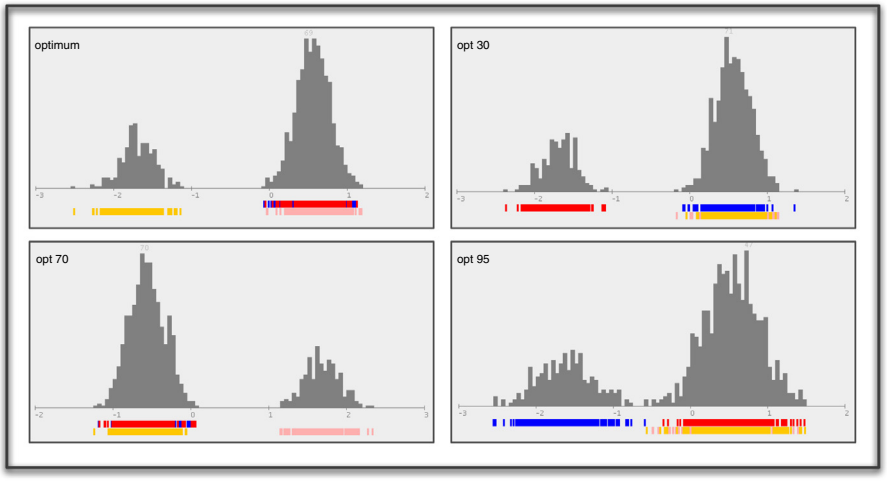


**Figure 8** Normal4 simulated data: histograms corresponding to the global optimum (top left) and local optima (top right and bottom plots) for the discriminant index using Tribes.

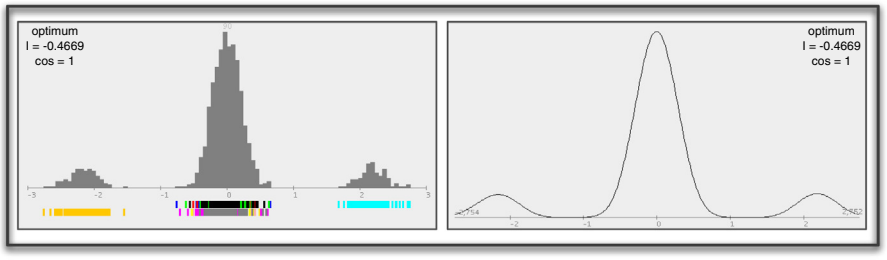


**Figure 9** Normal10 simulated data: histograms (left) and kernal estimators (right) corresponding to the global optimum for the discriminant index using GA.

particles for PSO does not need to be large. For larger data sets like the olive, reliability and simulated data sets, these values are increased. The number of iterations has been obtained by carrying out some preliminary runs on each data set and checking the convergence of the indices. Table 2 summarizes these values for both methods GA and PSO. In order to make the results of GA and PSO comparable, the choice of the number of individuals/particles and iterations are set such that GA and PSO lead to the same number of fitness evaluations given in Table 2. Contrary to GA and PSO, the number of particles varies in Tribes method which leads to no control of the number of fitness evaluations. The number of iterations is set to 100 after some preliminary runs on each data set and check of the convergence of the projection indices. We ran 100 times each optimization algorithm on the different data sets. We have to stress that the results would not be exactly the same for different values of the number of individuals/particles and iterations. But overall, when considering 100 runs (as we do), the method is quite robust in the sense that the structures detected in the five data sets are the same for different values of these parameters.

*6.2.1. Lubischew example*

Fig. 3 plots the ranked indices and the associated cosines of each projection vector with the "best" projection vector (associated with the optimum) using Tribes (left curves of each plot) and PSO (right curves of each plot) for the minimum kurtosis (top curves), the Friedman (middle curves) and the discriminant indices (bottom curves). For the first two indices using the two methods, the displays are quite similar. They reveal that for more than 70 runs over 100, the obtained projection vectors are the same (cosines equal to 1) but are different for the other 30 runs (cosines different from 1) which means that there are at least two potentially interesting views of the data. This result is well shown in Tribes method through the two landings of the index and the cosine curves. For the discriminant index, the curves are quite different. The values of the index decrease much faster with PSO than Tribes. When looking at the cosines curve, there is much more variability with PSO than Tribes which leads us to look at 3 or 4 projections. Recal that the large dot on the curves corresponds to the maximum of the indices over the 100 runs while the vertical line corresponds to the objective value of a second best projection that has been selected (see for example run 83 on top left curve).

The histograms of the distribution of the projected data corresponding to the maximum (resp. the second selected maximum) of the minimum kurtosis, Friedman and discriminant indices are displayed on the left (resp. right) of Fig. 4 with Tribes method. We notice that the three indices do not lead

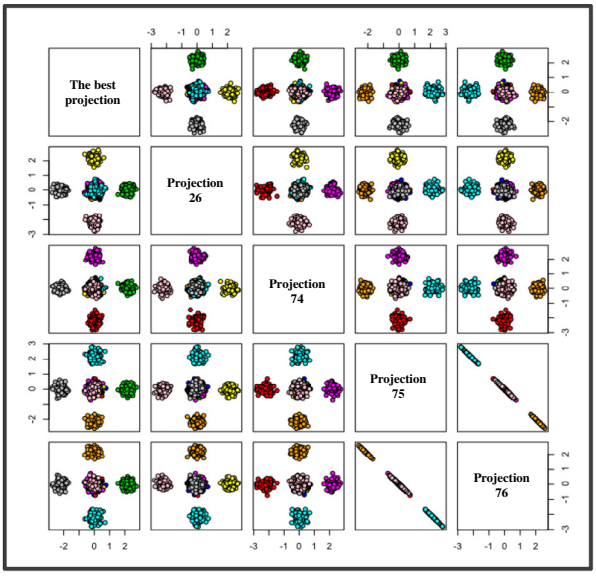


**Figure 10** Scatter plot matrix for Normal10 simulated data (var1 = the best projection, var2 = projection associated with the 26th local optimum, var3 = projection associated with the 74th local optimum, var4 = projection associated with the 75th local optimum, var5 = projection associated with the 76th local optimum).

to the identification of the same cluster(s). On the "best" projection (left histograms of Fig. 4), the Friedman index and the minimum kurtosis (resp. the discriminant) index detect the third (resp. the second) cluster as different from the other two clusters. On the second "best" projection (right histograms of Fig. 4), the minimum kurtosis and the Friedman indices detect the second cluster while the discriminant index detects the first cluster with Tribes and the second one with GA and PSO. No cluster structure is detected using the Friedman–Tukey and the maximum kurtosis indices. This result confirms the fact that these indices are more adequate to detect outliers than clusters.

The result of the index and the cosine curves (resp. the interesting structures) obtained with GA (resp. GA and PSO) on this example are very similar to the ones presented with PSO (resp. Tribes except the projection obtained above for the discriminant index) and are omitted.

From this result, we conclude that clusters are identified with the Friedman, the minimum kurtosis and the discriminant indices. Moreover, this example suggests the use of different indices in a complementary manner for larger or more complex data sets.

### 6.2.2. Olive data

For Olive data, the curves of the cosine and ranked values for the kurtosis index using PSO and GA (resp. Tribes) yield one landing (resp. three landings) which leads to one (resp. at least three) potential interesting projection (s).

The Friedman and the discriminant indices give the same curves as the kurtosis index with PSO and Tribes methods but with GA these indices provide curves with at least three landings (the plots are not displayed).

Fig. 5 displays three interesting projections corresponding to different local optima for the kurtosis index with the Tribes method using histogram graphics. On these plots the data are split in two parts which correspond to different regions for the oils. To be more precise, when looking at the histogram of the left plot on Fig. 5, the group on the right corresponds to olive oils from the southern region of Italy which contains 3 areas while the group on the left corresponds to olive oils from Sardinia and Northern Italy and contains 2 areas (see Table 1). GA and PSO methods give the same projection. The middle and right plots correspond to other projections; they lead us to detect another group structure.

Note that it is also possible to identify other clusters by considering local optima of the other projection indices such as the Friedman–Tukey or the discriminant indices.

Another possibility to identify other interesting structures (clusters) is to apply data selection. The left part of the best projection (associated with the optimum, see the left plot) presented on Fig. 5, is selected and analyzed using EPP-Lab. Fig. 6 displays the 100 ranked indices (top left) and the associated cosine of each projection vectors with the best projection vector (top right), and histogram (resp. kernel estimator) of the best projection (middle left, resp. middle right) and another projection (bottom left, resp. bottom right) for the kurtosis with GA. The index and cosine curves indicate the existence of at least two interesting projections. When looking at the histogram and the kernel estimator corresponding to both the optimum and local optimum, the data are split in two groups. For more detail, the projections come to separate the Northern region of Italy from Sardinia (see the projection 78 on the right).

### 6.2.3. Simulated data

In Fig. 7 we plot the 100 ranked values for the minimum kurtosis index with the "best" projection vector using the simulated data with GA (left curves), PSO (middle curves) and Tribes (right curves). While the PSO method leads to small variability of the projection index values for the one hundred launches, Tribes and GA supply different local optima.

For Normal4, the PSO method does not yield several landings and so does not allow to detect the four clusters. On the contrary, the Tribes and GA methods give several local optima corresponding to the three landings of the left and right curves in Fig. 7.

For Normal10 data set, the left and right curves (for GA and Tribes) are composed of several landings which correspond to different local optima susceptible to reveal interesting structures.

For these particular examples, the Friedman and the discriminant indices give the same curves (for the index and the cosine values) as the kurtosis index using GA, PSO and Tribes optimization methods except the Friedman index which gives curves with two landings using PSO for Normal10.

In Fig. 8 we visualize four interesting projections corresponding to the global optimum on top left and different local optima on top right and bottom plots of the discriminant index using Tribes method and Normal4 simulated data.

In each plot of Fig. 8, the data are clearly structured in groups and one of them corresponds to one of the three known clusters. Considering the four plots corresponding to the global and local optima of the discriminant index, Tribes succeeds

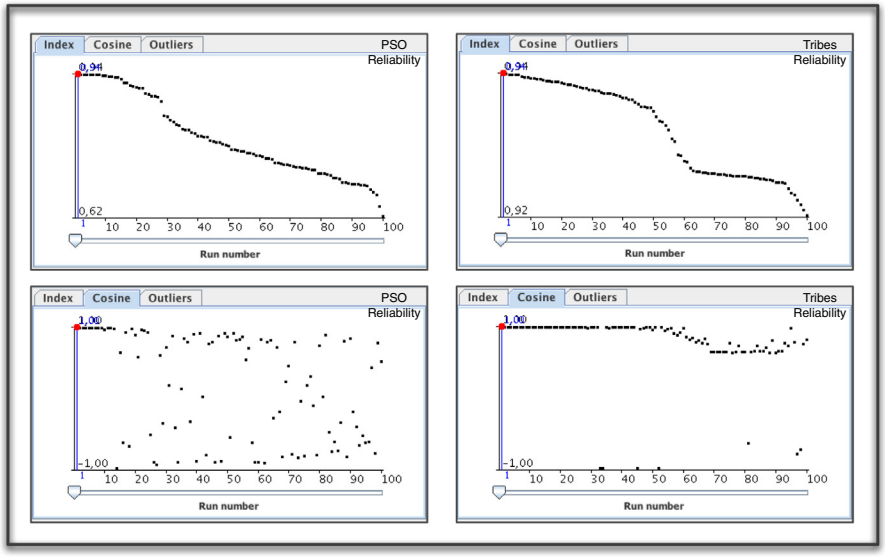


**Figure 11** Reliability data: plots of the ranked indices (top curves) and the associated cosines (bottom curves) for the Friedman index with PSO (left) and Tribes (right) methods.

to visualize the four clusters of Normal4 data set on one dimension. The kurtosis and the Friedman indices give projections that separate the data in clusters with Tribes, GA and PSO. The discriminant index with GA gives the same results as those presented with Tribes but when using PSO, the result is like the one achieved by the kurtosis index.

For Normal10 data set, Fig. 9 presents interesting projection corresponding to the global optimum of the discriminant index using GA. As seen, the data are divided in three parts. The right and the left parts contain one cluster while the middle part is composed of different colors which correspond to different clusters defined in the Normal10 simulated data. Through the global and local optima of the discriminant index, eight clusters among 10 are detected (not shown in this article). The Friedman index with the three methods and the discriminant index with Tribes give the same result presented on Fig. 9. The discriminant index with PSO only discovers four clusters through two projections associated with the optimum and the one local optimum. For the kurtosis index, the obtained projections consist of three parts where each one contains at least two groups using the three optimization methods.

The purpose of using the cosine notion is to identify different projections having the same index value. In some cases, this notion is not sufficient because we may have two different projections having the same index and cosine values.

Furthermore, when exploring the data set, we do not have prior knowledge of its structure (contrary to our examples), thus we cannot know whether these local optima give different projections or not. For this, we can use a scatter plot matrix for further analysis. This technique shows the relations among the variables in the data and can be used for any number of variables.

Now, in order to go further in the analysis of the EPP-Lab results, we use the Normal10 data set with 5 variables, which means the 4 projections (that detected the eight clusters) associated with the global optimum and three local optima (numbers 26, 74 and 75) and another projection associated with the local optimum number 76. This local optimum gives the same projection as that associated with the 75th local optimum.

Fig. 10 shows that the first four projections (that detected the eight clusters) are different contrary to projection 75 which is like the projection 76 (see the scatter plot : moderately strong (negative) correlations between projection 75 and 76). So, this confirms our previous remarks.

We conclude that the three methods propose several local optima which represent various interesting projections and highlight the complex structure hidden in the Normal4 and Normal10 data sets.

### *6.2.4. Outlying observations detection*

The objective of this study is to demonstrate the efficiency of EPP-Lab in detecting atypical chips set. Because of the high number of dimensions of the reliability data set, we focus on the kurtosis index that is fast to compute and dedicated to the detection of outliers (Peña and Prieto, 2001) and the Friedman index that is also fast to compute. The automatic procedure that leads to the identification of a certain percentage of outliers is explained in the above section.

To save space, we only show the results of the Friedman index with PSO and Tribes but we discuss the results of the rest of the indices and the optimization methods.

The index and the cosine curves of the Friedman index with PSO (left) and Tribes (right) on Fig. 11 show that there are at least two interesting projections. The kurtosis index gives the same result as the Friedman index with PSO and Tribes. For GA method, the Friedman index leads to the same curves than those found using the kurtosis index.

Fig. 12 displays projections obtained by Tribes using the Friedman index. A few chips are very far from the main bulk of the data when one looks at the histograms of the selected projections (the outliers are colored in orange). These high technology chips may be considered as outlying and may be taken out of production. In order to determine all outlying observations detected on one projection at each run, we plot the outliers graphic as shown in Fig. 13.

We recall that the $x$-axis represents the number of runs and y-axis includes the number of observations. At each run and when the observation is outlying, it is represented by a black

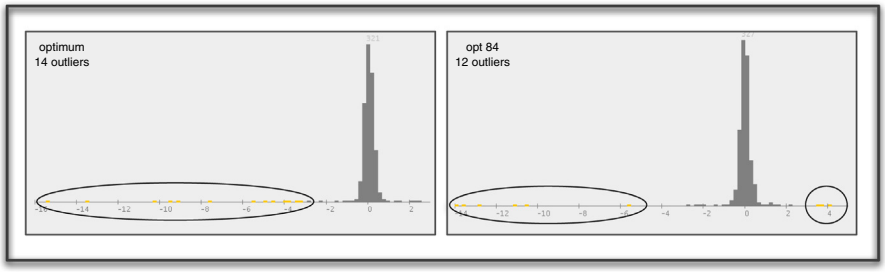


**Figure 12** Reliability data: histogram corresponding to the global optimum (left figure) and a local optimum (right) for the Friedman index using Tribes.

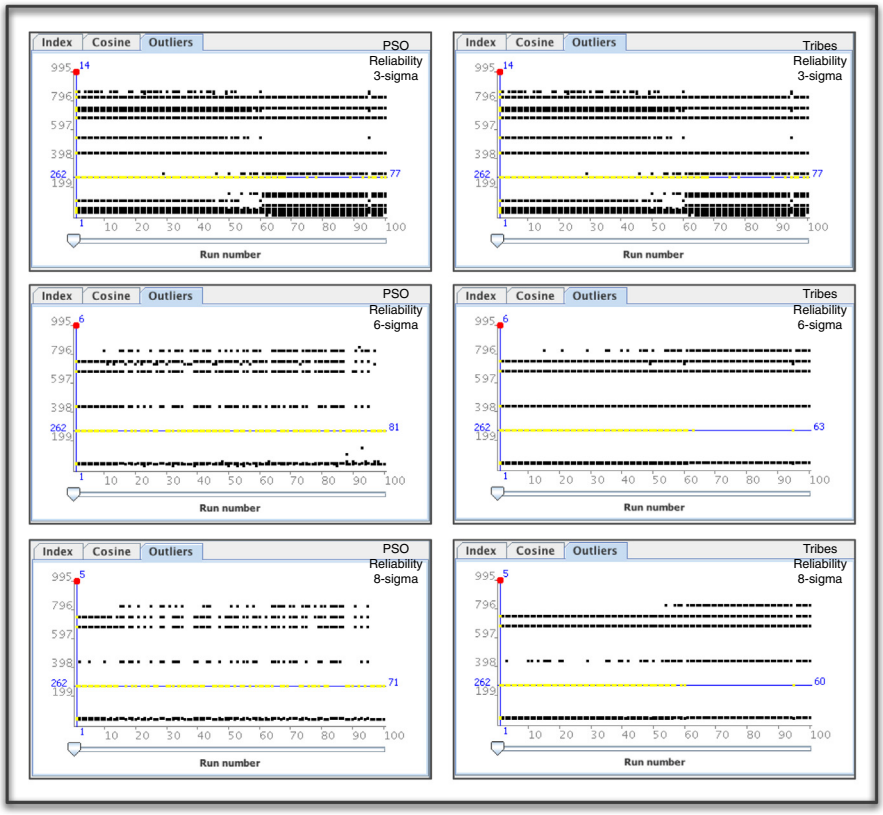


**Figure 13** Reliability data: outliers curves of 3-sigma (resp. 6-sigma and 8-sigma) principle (top, resp. middle and bottom) for the Friedman index using PSO (left curves) and Tribes (right curves).

dot on the graphic. In order to identify a percentage of outliers, we use a $k$-sigma criterion as defined in the above section. We remember that the parameter $k$ is set by the user. Fig. 13 presents the outliers curves of 3-sigma (resp. 6-sigma and 8-sigma) principle (top, resp. middle and bottom) for the Friedman index with PSO (left curves) and Tribes (right curves). The blue vertical line indicates the number of outliers obtained on the selected run (14 outliers for the first run on top curves) while the blue horizontal line points the number of times the selected observation is discovered as outlying throughout the 100 runs. We observe that when the parameter $k$ increases the number of outliers becomes smaller. Table 3 summarizes outlying observations and the number of times that have been discovered as such (see frequency column) at each fixed-parameter $k$ for the Friedman index using Tribes optimization method. We notice that the most outliers found by testing the kurtosis index with GA were found using the Friedman index with Tribes and in particular the observation number 262 even setting the parameter $k$ to 8. Arguably, both indices with the three methods give eventually and roughly the same result. The observation 262 is detected in all cases. The result is the same using the Friedman–Tukey and the discriminant indices with the three methods. This result is interesting because it is favorably compared with the univariate detection methods used in the industry which does not lead to the detection of observation 262. These univariate methods are based on the

k-sigma principle applied to the original variables and does not take into account the multivariate structure of the data set. Coherently, the company tries to detect as few as possible defective chips and with complete assurance. EPP-Lab is an interesting tool that meets their requirements. Indeed, first, it provides a rule based on k-sigma principle to flag an observation as outlying. The parameter k, as we noted earlier, is adjustable and therefore the company can freely choose the number of atypicals that suits their objectives. Secondly, running the optimization algorithm several times allows us to assert that the observation detected more than 50 times over 100 launches is really outlying, especially if it has been discovered by several methods and projection indices.

### *6.3. Comparison studies*

#### *6.3.1. Clustering techniques*

In this section we tested K-means and PCA methods (using R software) on Lubischev, olive and Normal4 data sets and compared the obtained results with those obtained by EPP-Lab.

**Table 3** Outlying observations and the number of times that have been discovered over 100 runs by setting the *k*-sigma principle for the Friedman index with the Tribes methods.

| Standard deviation | Observation number | Frequency |
|---|---|---|
| $3\sigma$ | 3 | 36 |
| | 19 | 100 |
| | 33 | 76 |
| | 39 | 51 |
| | 43 | 100 |
| | 48 | 40 |
| | 53 | 56 |
| | 74 | 39 |
| | 100 | 53 |
| | 134 | 39 |
| | 154 | 44 |
| | 262 | 77 |
| | 288 | 47 |
| | 433 | 100 |
| | 538 | 56 |
| | 678 | 100 |
| | 727 | 60 |
| | 744 | 100 |
| | 816 | 99 |
| | 841 | 4 |
| | 856 | 36 |
| $6\sigma$ | 33 | 62 |
| | 43 | 100 |
| | 262 | 63 |
| | 433 | 100 |
| | 678 | 100 |
| | 727 | 3 |
| | 744 | 100 |
| | 816 | 56 |
| $8\sigma$ | 33 | 59 |
| | 43 | 96 |
| | 262 | 60 |
| | 433 | 65 |
| | 678 | 100 |
| | 744 | 99 |
| | 816 | 43 |

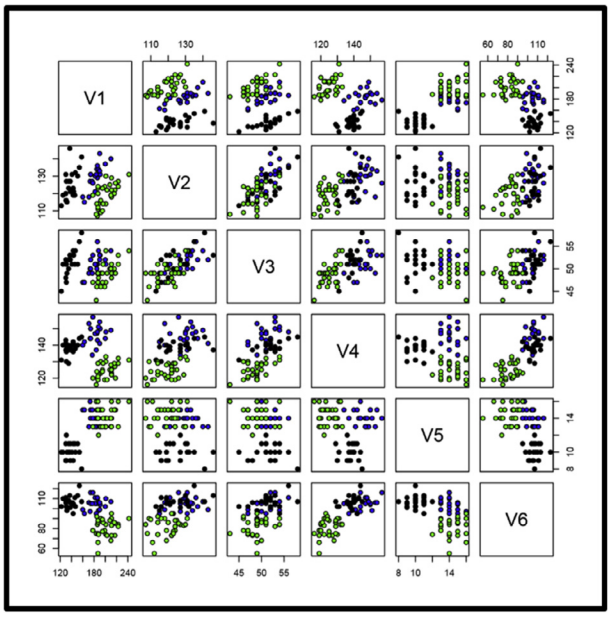


**Figure 14** Scatter plot for Lubischev data.

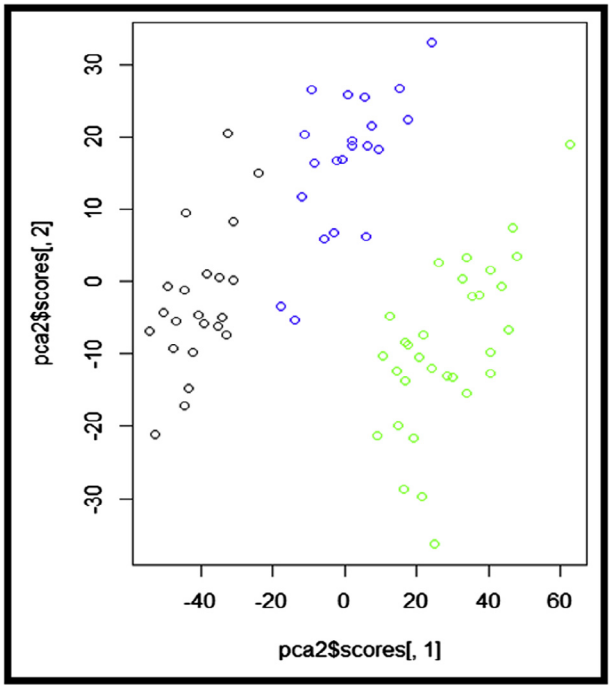


**Figure 15** Lubischev data set using the first (left) and second (right) axes of PCA.

In order to show the dispersion of the Lubischev's clusters, Fig. 14 displays the scatter plot with the same representation type used in EPP-Lab (the colors represent the same clusters shown in EPP-Lab). *K*-means discovered the clusters 1 and 2 with an accuracy of 80%, the cluster 3 with an accuracy of 100%. PCA is not capable to well discover the three clusters as displayed in Fig. 15.

Olive data set has been processed in various supervised and unsupervised clustering techniques. For example, Caussinus

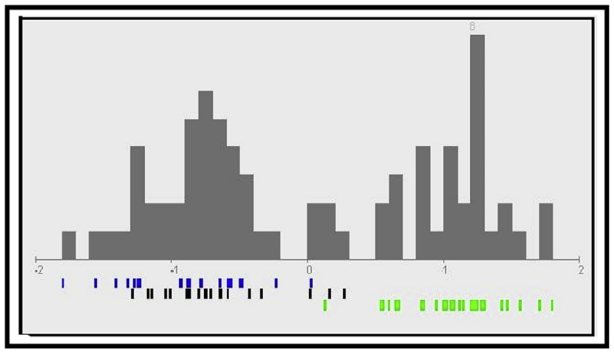

**Figure 16** Lubischew example: histograms of the distribution of the projected data on the "best" projection with PSO for the Friedman index – No convergence case.

and Ruiz-Gazen (2007) showed that PCA is far from giving the above result, PCA is not able to separate the clusters. This result has also been confirmed in Hou (2012) where the author addressed a detailed comparison between PCA and two-dimensional Projection Pursuit using the kurtosis index and quasi-power optimization method. The results reported using the two-dimensional Projection Pursuit are similar to those displayed in Fig. 5 (see the interesting projection associated with the optimum for the kurtosis index).

For Normal 4, nor *K*-means neither PCA are capable to determine the clusters. In fact, *K*-means detects four clusters of size (500, 114, 250, 136) respectively but they are not the true clusters of Normal4 because: 1 – the true size of each cluster is 250 and 2 – the content of each cluster doesn't seem correct. PCA yields the same results as *K*-means.

### *6.3.2. Convergence of the projection index*

Generally, the optimization methods require a suitable number of iterations to allow the objective function to converge toward the optimum. It is important to show the impact of choosing an inadequate number of iterations. Let show you one case of no convergence of PSO method with the Friedman index when setting the number of iteration to 20 (instead of 50). Fig. 16 shows the interesting projection associated with the optimum value for the Friedman index using PSO. The optimum found in this case is far from the optimum found previously which clearly finds the clusters. This means that PSO needs more iterations to reach the optimum. EPP-Lab provides a convergence projection index curve that shows to the user the stability of the projection index while it is running. This feature can be useful in helping the user to set the iterations number.

## 7. Computation time and limitations

To estimate the limits of our application, an experimental plan is performed. Simulated data sets are generated with different sizes: a number of observations $N$ between 40 and 100,000 and a number of dimensions $P$ between 2 and 50. The tested indices are Friedman, kurtosis and discriminant and the tested methods are GA, PSO and Tribes. The experiments were performed on a Dell Precision 2.54 GHz and each experiment was restarted 100 times. In Fig. 17, the $x$-axis represents the number of observations $N$ and the $y$-axis represents the computation time (in seconds) of 100 runs.

Regarding the computational times, the kurtosis is always the fastest index while the discriminant and the Friedman indices are slower. This can be explained because the indices have a different computational complexity. For the largest processed data set (100,000 observations and 50 variables), the kurtosis takes only three hours to complete the 100 runs with the PSO method. We remark on the left of Fig. 17 that the kurtosis index does not exceed 10 s to perform 100 runs on a data set containing 1000 observations and 10 variables while the Friedman index takes about 100 s to calculate the same data set. As for the discriminant index, its computational time is greater than 100 s for a data set with only 100 observations and 10 variables.

Concerning the Friedman index, the computation times are slightly longer than the kurtosis index, but it is still very acceptable. On the other side, the discriminant index cannot be used for large data sets because it is very slow. In fact, it even had to be removed from the experimental plan because it exceeded the maximum allowed time (3 h). For example, with the PSO method, it took over 3 h and 30 min to calculate the 100 runs of a data set containing only 1000 individuals and

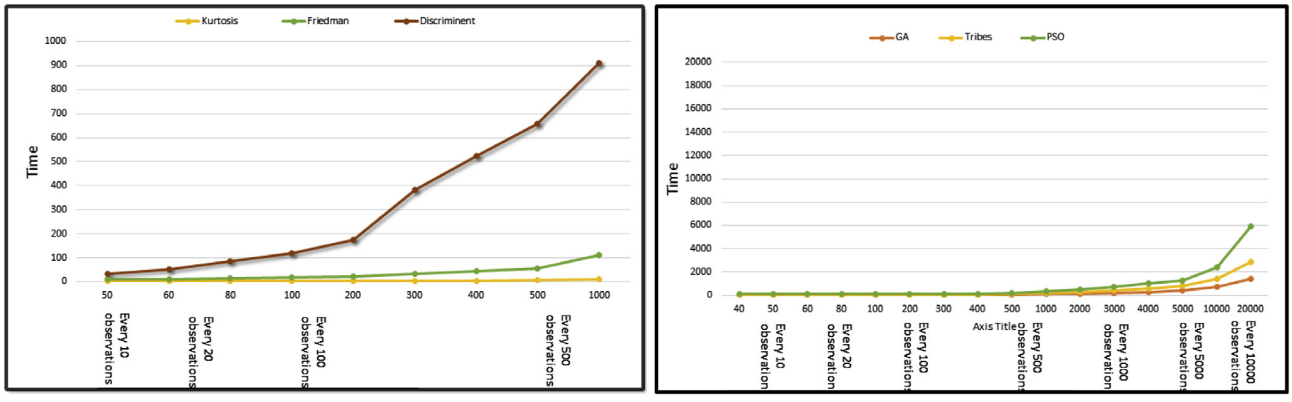


**Figure 17** Comparison of computation times of the different indices (resp. methods) on the left (resp. right) using PSO methods (resp. Frieman index) ($P = 10$ and $N$ between 40 and 1000, resp. $P = 10$ and $N$ between 40 and 20,000).

10 variables. With the same method, the kurtosis index has taken only 10 s and the Friedman index 1 min and 40 s.

The PSO method is the slowest of the three methods but the computation times are acceptable for the user. Although the number of evaluations of the objective function is not the same for the three methods (because the number of particles in the Tribes method is variable), we observe that Tribes is faster than PSO. In our implementation, GA is approximately 10% faster than Tribes and 35% faster than PSO. For small data sets, the computation times are negligible. Notice however that in a context of large data sets, our application is suitable for parallelization by simply considering a run as a task and distributing each task to a processor of a parallel computer.

## 8. Conclusion

EPP-Lab is a tool for finding several interesting projections that contain clusters and/or outliers in large and multidimensional data sets. We used bio-inspired optimization methods to optimize five projection pursuit indices. We validated the performance of the methods on real and simulated data sets that are structured in clusters or contain outliers. EPP-Lab achieves both targets: search and visualization of hidden structures in multidimensional data sets. The search is carried out through the use of several projection pursuit indices, several optimization methods and several runs. This strategy allows to find several local optima corresponding to various structures for a data set. The visualization is performed through the graphical interface of EPP-Lab that allows the user viewing all the associated projections for all local optima using histograms and kernal estimators. Moreover, EPP-Lab has various features that permit to the users to well analyze the large volume of the data and detect the hidden patterns.

As stated above, the code of EPP-Lab is available online. One of the perspectives is to implement EPP-Lab in R software and make it as a package of EPP method. In addition, EPP-Lab is a modular software, it is easily possible to add new mechanisms for example feature selection methods to reduce the dimensionality of a data set. Furthermore, it would be interesting to study the stopping tests of the number of iterations for the detection of outliers.

## Acknowledgment

The author wishes to acknowledge Prof. Anne Ruiz Gazen and Dr. Alain Berro for their supervision to perform this work, and their constructive and creative ideas. Dr. Alain Berro has a great contribution in implementing this software. The author also thanks Emilie Chabbert and Ingrid Griffit for their contribution. A great thanks to the reviewers who highly contributed to the improvement of this work.

## References

Achard, V., Landrevie, A., Fort, J., 2004. Anomalies detection in hyperspectral imagery using projection pursuit algorithm. In: Image Signal Processing for Remote Sensing X, vol. 5573. SPIE, pp. 193–202.

Alghamdi, H., Selamat, A., Abdul Karim, N., 2014. Arabic web pages clustering and annotation using semantic class features. J. King Saud Univ. Comput. Inf. Sci. 26, 388–397.

Aljumah, A., Ahamad, M., Siddiqui, M., 2013. Application of data mining: diabetes health care in young and old patients. J. King Saud Univ. Comput. Inf. Sci. 25, 127–136.

Baker, N., 1991. The Detection of Outliers in Laboratory Safety Test Datasets. Master of Science in Statistics.

Berro, A., Larabi Marie-Sainte, S., Ruiz-Gazen, A., 2010. Genetic algorithms and particle swarm optimization for exploratory projection pursuit. Ann. Math. Artif. Intell. 60, 153–178.

Caussinus, H., Ruiz-Gazen, A., 2007. Selected Contributions in Data Analysis and Classification, chapter Classification and Generalized Principal Component Analysis. Springer-Verlag, Berlin Heidelberg.

Caussinus, H., Ruiz-Gazen, A., 2009. Exploratory data analysis, chapter Exploratory projection pursuit. G. Govaert.

Clements, A.M., Jones, M., 1991. An ecological example of the application of projection pursuit to compositional data. Vegetatio 95, 101–107.

Clerc, M., 2006. Particle Swarm Optimization. International Scientific and Technical Encyclopaedia Wiley, Hoboken.

Crawford, S., 1991. Genetic optimization for exploratory projection pursuit. pages 318–321. Interface Foundation of North America. Conference Theme: Critical Applications of Scientific Computing: Biology, Engineering, Medicine, Speech., ElaineM. Keramidas. In: Proceedings of the 23rd Symposium on the Interface.

Faith, J., Brockway, M., 2006. Targeted projection pursuit tool for gene expression. J. Integr. Bioinf.

Fogel, L.J., Owens, A.J., Walsh, M.J., 1966. Artificial Intelligence Through Simulated Evolution. Wiley, New York.

Friedman, J., 1987. Exploratory projection pursuit. Am. Stat. Assoc. 82 (1), 249–266.

Friedman, J., Tukey, J., 1974. Exploratory projection pursuit. A Projection Pursuit Algorithm for Exploratory Data Analysis 23, 881–889.

Ghodami, S., Larabi Marie-Sainte, S., 2015. Support vector machine based exploratory projection pursuit optimization for user face identification. In: Proceedings of IEEE International Conference on Information and Communication Technology Research (ICTRC2015), pp. 4799–8966.

Goswami, B., Mandal, D., 2013. A genetic algorithm for the level control of nulls and side lobes in linear antenna arrays. J. King Saud Univ. Comput. Inf. Sci. 25 (2), 117–126.

Hogge, V., Austin, J., 2004. A survey of outlier detection methodologies. J. Artif. Intell. Rev. 22, 85–126.

Holland, J.H., 1975. Adaptation in Natural and Artificial Systems. Systems, University of Michigan Press, Ann Harbor.

Hou, S., 2012. Improved projection methods for exploratory data analysis in chemistry (Technical report).

Huber, P., 1985. Projection pursuit. Ann. Stat. 13 (2), 435–475.

Jones, M., Sibson, R., 1987. What is projection pursuit? (with discussion). J. R. Stat. Soc. A 150, 1–36.

Kennedy, J., Eberhart, R., 1995. Swarm Intelligence. Yuhui Shi.

Klinke, S., 1997. What is projection pursuit? (with discussion). Data Struct. Comput. Stat. Physica-Verlag, 1–36.

Larabi Mari-Sainte, S., Berro, A., Ruiz-Gazen, A., 2010. An efficient optimization method for revealing local optima of projection pursuit indices. In: Proceedings of Swarm Intelligent. ANTS2010. Brussels, Belgium, pp. 60–71.

Lee, Y.D.C., Park, J., Lee, E., 2013. Pptree: projection pursuit classification tree. Electron. J. Stat. 7, 1369–1386.

Lubischew, A., 1969. On the use of discriminant functions in taxonomy. Biometrics 18, 455–477.

Malpika, J.A., Rejas, J.G., Alonso, M.C., 2008. A projection pursuit algorithm for anomaly detection in hyperspectral imagery. Pattern Recognit. 41, 3313–3327.

Martinez, W., Martinez, A., 2001. Computational Statistics Handbook with Matlab. CRC Press (Taylor and Francis Group).

Morton, S., 1989. Interpretable projection pursuit (Technical report).

Nason, G., 1992. Design and Choice of Projection Indices (Ph.D. thesis). CRC Press (Taylor and Francis Group), University of Bath.

Peña, D., Prieto, F., 2001. Cluster identification using projections. J. Am. Stat. Assoc. (456)

Posse, C., 1995a. Projection pursuit exploratory data analysis. Comput. Stat. Data Anal. 29, 669–687.

Posse, C., 1995b. Tools for two-dimensional exploratory projection pursuit. J. Comput. Graph. Stat. 4, 83–100.

Rechenberg, R.I., 1973. Evolutionsstrategie: Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution. Frommann-Holzboog, Stuttgart, Germany.

Rohatsch, T., Poppel, G., Werner, H., 2006. Projection pursuit for analyzing data from semiconductor environments. IEEE Trans. Semicond. Manuf. 19 (1).

Ruiz-Gazen, A., Berro, A., Larabi Marie-Sainte, S., 2010. Detecting multivariate outliers using projection pursuit with particle swarm optimization. In: Proceedings of Computation Statistics Compstat 2010. Springer-Verlag, pp. 89–98.

Schwefel, H.P., 1981. Numerical Optimization of Computer Models. John Wiley and Sons, New York.

Sevkli, M., Mamedsaidov, R., Camci, F., 2014. A novel discrete particle swarm optimization for p-median problem. J. King Saud Univ. Eng. Sci. 26 (1), 11–19.

Sun, J., 1993. Some practical aspects of exploratory projection pursuit. SIAM J. Sci. Comput. 14 (1), 68–80.

Swayne, D., Lang, D., Buja, A., Cook, D., 2003. Ggobi: evolving from XGobi into an extensible framework for interactive data visualization. Comput. Stat. Data Anal. 43 (4), 423–444.

Upadhyay, P., Kar, R., Ghoshal, S., 2014. A new design method based on firefly algorithm for IIR system identification problem. J. King Sasud Univ. Eng. Sci. (in Press, Corrected Proof).

Yenyukov, I.S., 1989. Indices for projection pursuit. Nova Science Publishers, New York, Data Analysis Learning Symbolic and Numeric Knowledge. Diday. pp. 181–188.